R reference card

General commands

Remove all the existing objects	rm(list=ls())
Install a package	install.packages(“<package_name>”)
Load a library	library(<library_name>)

Read a dataset from a .csv file	read.csv("<filename>.csv",
	header = <TRUE or FALSE>,
	na.strings = <NA character>,
	stringsAsFactors = <TRUE or FALSE>)

Write a dataset to a .csv file	write.csv(<dataset>, file = "<filename>.csv")
Print the structure of an object	str(<object>)
# of elements	length(<object>)
Combine objects into a vector	c(<object>,<object>,...)
Combine objects as columns	cbind(object, object, ...)
Combine objects as rows	rbind(object, object, ...)

Operations

transform a character variable into a number	as.numeric(<variable>)
transform a character variable into a date	as.Date(<variable>)
create a dataframe	data.frame(<var1>, <var2>, …, <varN>)
transform a data frame into a matrix	as.matrix(<data frame>)
create a new variable	<dataset>$< variable> <- <variable>
bind variables together	cbind(<variable_1>, <variable_2>)
Create first differences	diff(<variable>)

Create an all-to-all merge between two datasets	merge(<dataset_1>, <dataset_2>,
	by="<variable_name>",
	all=TRUE)

Merge two datasets by more than one variable	merge(<dataset_1>, <dataset_2>, by = c("<var1 name>", "<var2 name>"))
Print summary information	summary(<data object>)
Create a sequence	seq(<start #>, <end #>, by = <step size>)

Create a new function	<functionname> = function(<input variable>){
	<operations>
	}

Choose rows	<dataset>$<variable>[condition]
Assign operator	<-
Equal operator	==
Not equal operator	!
And operator	&
Or operator	|
Assign NA	<dataset>$<variable> [<row>] <- NA
Remove rows with NA	<dataset> <- na.omit(<dataset>)
Keep data rows that satisfy condition	filter(<dataset>, condition)
Create a unique list of values	unique(<variable>)
Calculate percentiles	quantile(<variable>, <list of percentiles>, na.rm=TRUE)
Round a value to a # of decimals	round(<var_list>, #decimals)

Dates

Transform a character variable into a date	as.Date(<variable>)
Extract year	as.numeric(substr(<date var>, 1, 4))
Extract month	as.numeric(substr(<date var>, 5, 6))

Graphics

Create a line plot	plot(<x-variable>, <y-variable>, type = 'l')
Create a histogram	hist(<variable>, breaks=<# bins>)
Create a Q-Q plot vs. normal	qqnorm(<variable>)
Add the base-line to the Q-Q plot	qqline(<variable>)
Add a straight line	abline(<intercept>, <slope>)
Add a regression line	abline(<regression object>)

Plot options

Title	main="<Plot name>"
x-label	xlab="<x label>"
y-label	ylab=”<y label>”
line type	lty = <number>
point type	pty = <number>
color	color = “<color name>”

Regression

Run a regression	lm(y ~ x1 + … + xN, data = <data frame>)
Choose a base case for a dummy variable	<dataset>$<dummy> = relevel(<dataset>$<dummy>, ref = <base case #>)
Create a regression object (robj)	<robj> <- lm(…)
Summary data from regression	summary(<robj>)
Extract the residuals from a regression object	resid(<robj>)
Extract R^2 from a regression object	summary(<robj>)$r.squared
Extract alpha coefficients	coef(summary(<robj>))[“(Intercept)”]
Extract beta coefficients	coef(summary(<robj>))[“<var. name>”]
Extract alpha standard error	coef(summary(<robj>))[“(Intercept)”, “Std. Error”]
Extract beta standard error	coef(summary(<robj>))[“<var. name>”, “Std. Error”]
Extract t-statistic for alpha	coef(summary(<robj>))[“(Intercept)”, “t value”]
Extract t-statistic for beta	coef(summary(<robj>))[“<var. name>”, “t value”])

Calculate confidence intervals	predict.lm(<robj>,
	newdata = <prediction dataset>,
	level = <conf. level>,
	interval = "<prediction type>")

Group operations

Calculate function by group	aggregate(<variable>,
	by=list(group),
	FUN = <function name>)

dplyr package

Load the ***dplyr*** library	library(dplyr)
Group data	<grouped data> <- group_by(<data frame>, <variable>)

Run a regression by group and save results	<group reg obj> <- <grouped data> %>%
	do(<reg_obj> = lm(y ~ x1 + … + xN,
	data = .))

Extract list of R^2 from a group regression object	summarise(<group reg obj>, r2 = summary(reg_obj)$r.squared)
Order data by group, based on variable values	arrange(<dataset>, <variable>)

quantmod package

Load the ***quantmod*** library	library(quantmod)
Download data	getSymbols(c('<ticker>'),src='<source>');